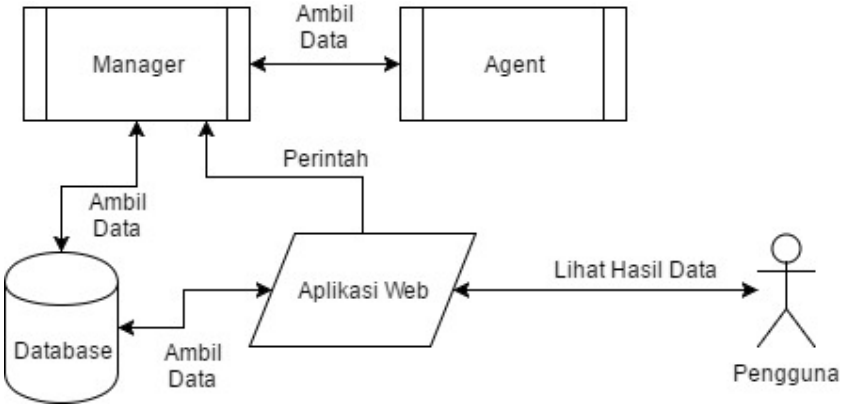
BAB 4 PERANCANGAN

4.1 Deskripsi Umum Sistem

kehidupan sehari-hari tak lepas dari pemanfaat Alat teknologi. Disini dilihat dengan banyaknya pemanfaat alat-alat teknologi yang meningkat salah satunya alat bernama raspbberry pi Karena kemampuan Raspberry Pi yang sangat baik dalam berintegrasi dengan berbagai sensor dan perangkat tambahan lain. Tetapi saat penggunaan salah alat IoT ini masing belum adanya pengawasan nilai *monitoring* pengawasan perangkat IoT itu sendiri. Pemanfaat untuk dapat melakukan aktifitas secara maksimal pada perangkat *Internet Of Thing* adalah dengan membuat sistem *monitoring* perangkat IoT. Sistem *Monitoring* perangkat IoT berbasis fitur Statisik merupakan sistem yang digunakan untuk mengetahui kondisi perangkat IoT secara efektif, sehingga dalam proses *monitoring* perangkat IoT dengan menggunakan mikrokomputer Raspberry Pi akan dapat lebih mudah pengolahan datanya dan menampilkan informasi yang diperlukan.



Gambar 4.1 Deskripsi Umum Sistem

Beberapa komponen yang terkait Antara lain *Manager, Agent , Database* dan web. Manager bertugas untuk mengambil data dan diproses oleh agent, kemudian agent akan melakukan pekerjaannya sesuai dengan perintah yang diterimanya. Kemudian hasil *monitoring* akan disimpan oleh manager dalam database. Setelah itu dengan web data hasil *monitoring* yang tersimpan tersebut dapat ditampilkan.

4.2 Analisis Kebutuhan Fungsional dan Kebutuhan Non-Fungsional

Dalam analisis kebutuhan fungsional dan non-fungsional in bertujuan untuk menggali kebutuhan yang diperlukan dalam membangun system *monitoring* untuk perangkat IoT. Analisis kebutuhan tersebut dijelaskan pada sub bab berikut.

4.2.1 Kebutuhan Fungsional

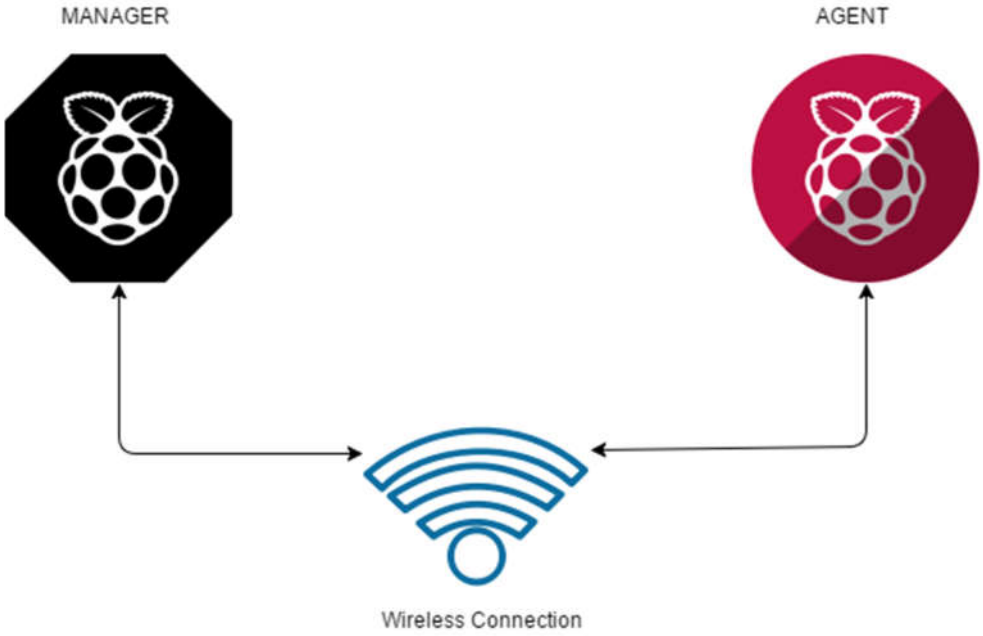
Kebutuhan yang pertama adalah kebutuhan fungsional yang menjelaskan apa saja yang harus dilakukan sebuah system sehingga pengguna bisa menggunakannya. Kebutuhan ini juga mendefinisikan tentang fungsionalitas system apakah fungsionalitas system sudah sesuai dengan yang diharapkan.

4.2.2 Kebutuhan Non-Fungsional

Kebutuhan non-fungsional merupakan kebutuhan yang tidak terhubung secara langsung dengan layanan system. Kebutuhan non-fungsional dapat diartikan sebagai Batasan yang ada pada pengembangan system seperti yang tertera pada Batasan masalah yaitu Memodifikasi MIB dengan kategori OID, Pengelolahan strutkur data Json dengan Dictionary, Sistem *monitoring* perangkat IoT diakses menggunakan aplikasi web.

4.3 Perancangan Topologi Jaringan

Perancangan topologi Jaringan, jaringan dibuat dengan kondisi terdapat 1 wireless connection, 1 raspbberry pi sebagai *Agent* dan 1 raspbberry pi sebagai *Menager*. Topologi pada Gambar 4.2 digunakan penulis dalam melakukan implementasi dan pengujian pada penelitian ini.

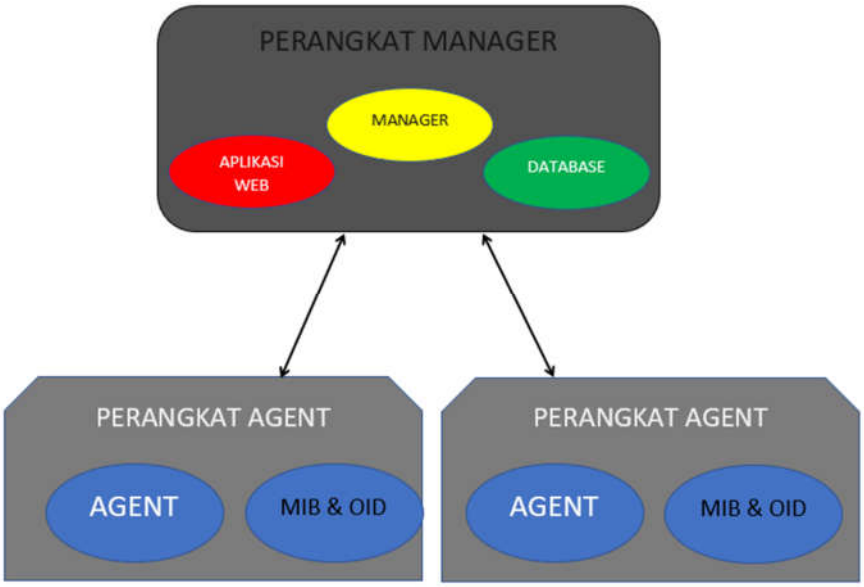


Gambar 4.2 Topologi Jaringan

Penjelasan Topologi:

1. 1 Agent terhubung dengan *Wireless connection*.
2. *1 Manager* terhubung dengan *Wireless connection*.
3. *Manager* Memonitor *Agent*
4. *Agent* Dimonitor oleh *Manager*

4.4 Perancangan Sistem *Monitoring* Perangkat IoT



Gambar 4.3 Perancangan Sistem *Monitoring* Perangkat IoT

Pada gambar 4.3 komponen-komponen yang terkait antara lain *Agent, Manager, Database* dan Web. Perangkat *Manager* terdiri dari *database*, aplikasi web dan script *manager*. Perangkat ini memilki fungsi untuk memerintah agent dalam pengambilan data *monitoring*. Perangkat *agent* terdiri dari script *agent* dan MIB yang akan diambil pada proses *monitoring*. Setelah itu, *Agent* mengirimkan data kepada Manager untuk diproses dan disimpan kedalam database. Kemudian data tersebut nantinya akan ditampilkan untuk memunculkan informasi melalui apilkasi web. Sistem *monitoring* perangkat IoT ini tujuan akhirnya yaitu untuk dapat digunakan mengumpulkan berbagai data dan dilakukan analisis. Sistem *Monitoring* untuk Perangkat IoT ini akan memantau beban trafik pada link jaringan. Sistem akan membuat halaman HTML yang berisi gambar yang mengambarkan *OID* melalui *monitoring* yang dibuat dengan waktu yang di inginkan.

4.4.1 Perancangan agent

Tujuan dari perancangan a*gent* adalah sebagai berikut:

1. Melakukan pengambilan data dari hasil *monitoring*.

   Jenis datanya berupa komponen untuk dianalisisi, pengambilannya menggunakan *library* atau kode program.

2. Untuk dapat merespon perintah dari *Manager*

   Agent merespon perintahi *manager* kemudian sekaligus mengirimkan data sesuai dengan perintah yang diterimanya.

3. Untuk melakukan pengiriman hasil *monitoring* ke *Manager*

   Agent bisa melakukan pengiriman data seperti yang dibutuhkan *manager*.

4.4.2 Perancangan Manager

Perancangan kommponen *Manager* ini digunakan agar *manager* dapat melakukan beberapa hal:

1. Melakukan pengiriman perintah untuk mendapatkan hasil *monitoring*

   Perintah yang diisi pengguna dapat direspon dan dikirimkan oleh manager, kemudian manager mencari dan mengakses perintah tersebut kedalam database. Setelah itu manager akan mengirimkan perintah tersebut kedalam agent.

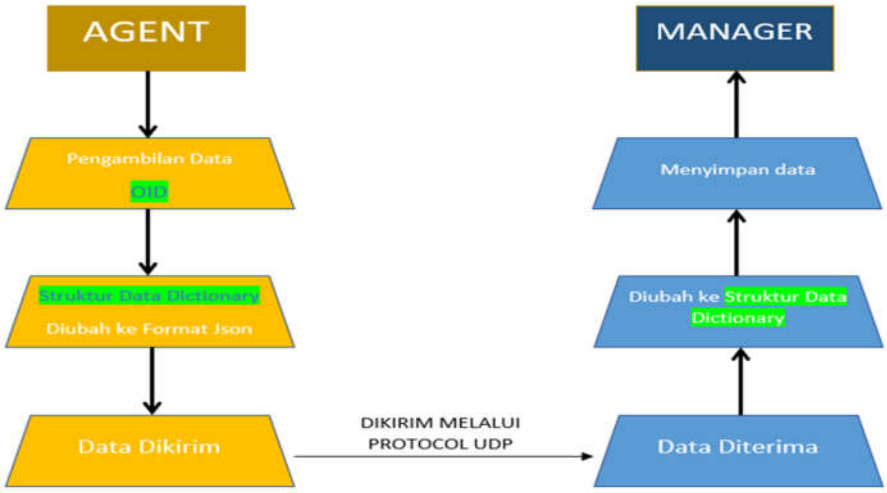
2. Melakukan penerimaan devices yang saling terkoneksi

   Manager bisa menerima devices yang saling terkoneksi, me*monitoring* status melalui devices lalu menyimpan data devices pada database.

3. Melakukan penerimaan hasil *monitoring*

   Pada langkah ini, hasil *monitoring* yang dikirimkan oleh *agent* diterima oleh manager dan disimpan kedalam database.

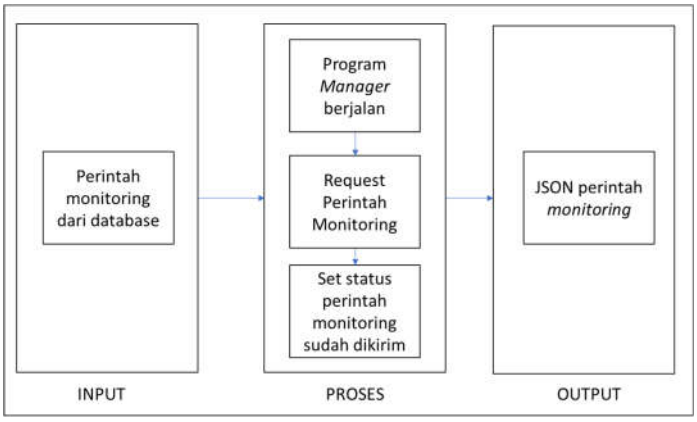
4.5 Perancangan pengiriman dan penerimaan data



Gambar 4.4 Perancangan Pengiriman dan Penerimaan Data

Gambar 4.4 itu Pertukaran data akan diubah terlebih dahulu ke dalam struktur data *Dictionary* kemudian di convert dalam bentuk JSON lalu dikirimkan melewati protokol jenis UDP. Kemudian melakukan pengubahan format JSON menjadi bentuk *Dictionary* kembali oleh program manager.

4.5.1 Pengiriman data dari manager

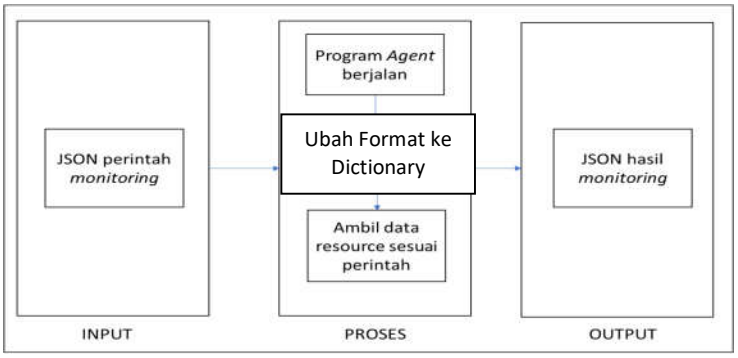


Gambar 4.5 Perancangan Pengiriman Perintah dari Manager

Proses pengiriman data pada bagian *manager* ditunjukkan pada Gambar 4.5. Dalam diagram dibawah inputnya yaitu perintah monitor yang tersimpan kedalam sebuah *database,* perintah ini didapatkan dari set perintah yang dimasukkan oleh pengguna*.* Setelah itu program *manager* melakukan *request* perintah dari *database*, hal ini dilakukan untuk mengambil perintah yang ada pada *database*, lalu *manager* melakukan *set* status perintah sudah dikirim pada perintah yang sudah diambil, perintah *monitoring* yang diambil masih dalam bentuk *dictionary* sehingga masih belum bisa dikirimkan ke *agent*, sehingga harus diubah menjadi format JSON terlebih dahulu. Setelah diubah menjadi format JSON maka perintah *monitoring* dapat dikirimkan. Hal ini dapat dilihat pada diagram alir bahwa *output* yang dihasilkan adalah *JSON* perintah *monitoring.*Setelah *manager* menghasilkan *output* berupa JSON perintah *monitoring* maka siap untuk dikirimkan pada *agent.*

4.5.2 Penerimaan data oleh agent

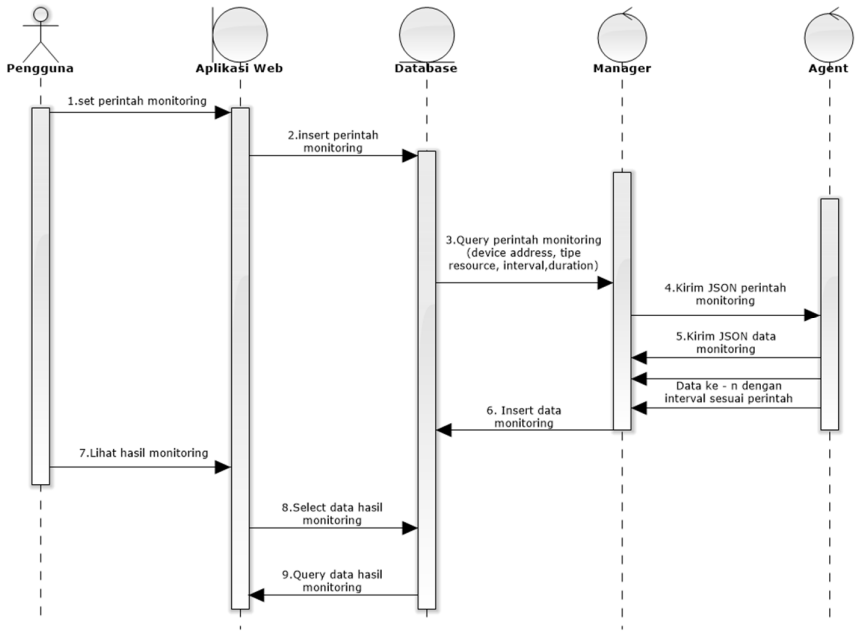
Pada bagian ini akan menjelaskan proses penerimaan data pada bagian *agent*, hal ini dapat dilihat pada diagram alir pada Gambar 4.6



Gambar 4.6 Penerimaan *Monitoring* Agent

Dalam Gambar 4.6 diatas dijelaskan manager meneria *input* yaitu data format JSON. Kemudian program *agent* melakukan pengubahan kedalam format *dictionary* untuk dilakukan pengolahan dan mengambil *resource* seperti perintah, *resource* yang diambil dapat berupa *resource device* atau *network activity*. Data yang diambil dijadikan ke dalam bentuk *dictionary,* data yang ada dalam bentuk dictionary belum dapat dikirim, sehingga harus diubah ke dalam format JSON. *Agent* menghasilkan *output JSON* hasil *monitoring* untuk dikirimkan kembali kepada *manager*.

4.5.3 Proses *monitoring*

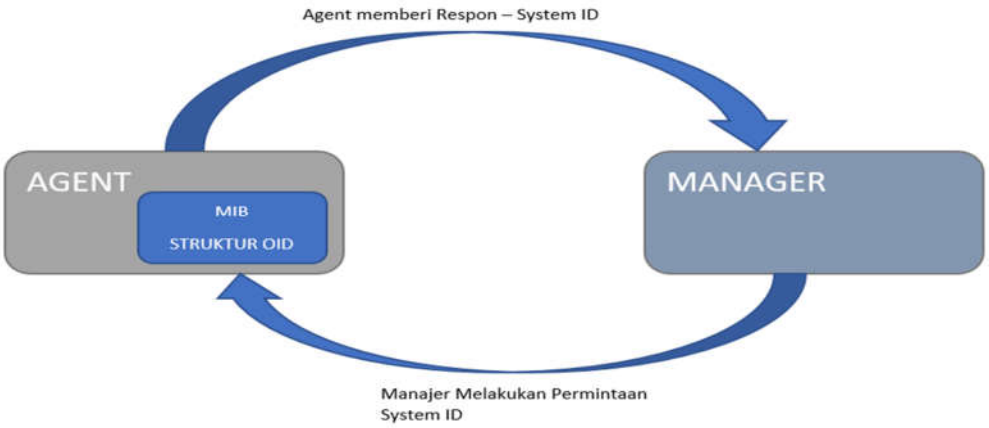


Gambar 4.7 Sequence diagram proses *monitoring*

Pada Gambar 4.7 ini akan dijelaskan proses *monitoring* data dan peran masing-masing komponen. Komponen yang terlibat pada diagram adalah pengguna, aplikasi web, database, manager dan agent. Pertama-tama pengguna melakukan proses set perintah *monitoring* melalui aplikasi web, setelah itu aplikasi web melakukan proses input perintah *monitoring* pada database. Kemudian manager mengambil data perintah *monitoring* dari database. Proses berikutnya adalah manager mengirim perintah *monitoring* yang sudah ada dalam format JSON ke agent. Agent yang menerima perintah *monitoring* akan mengambil resource sesuai perintah dan mengirim data yang dibutuhkan waktu dan interval yang sudah ditentukan. Setelah itu manager menerima data hasil *monitoring* dan menyimpan ke dalam database. Pengguna dapat melihat hasil *monitoring* melalui aplikasi web. Aplikasi web mengambil data hasil *monitoring* dari database dan akan ditampilkan dalam bentuk grafik.

4.6 Perancangan Struktur data OID

OID, *Object Identifier* adalah angka yang dibuat oleh MIB, *Object of Interest* dan *Instance*. Setiap *identifier* unik untuk perangkat, dan saat ditanya akan memberikan informasi tentang permintaan OID yang telah diminta.MIB adalah kumpulan pertanyaan yang bisa diajukan oleh *Manager* kepada *agent* tersebut. *Agent* mengumpulkan data ini secara lokal dan menyimpannya, seperti yang didefinisikan di dalam MIB. Jadi, *Manager* harus menyadari pertanyaan standar dan pribadi ini untuk setiap jenis agen. Agent dari perangkat yang diberikan, yang kemudian membagikan informasi terorganisir dari *database* yang dibuatnya dengan *manager*, yang selanjutnya menerjemahkannya menjadi laporan, grafik dan lainnya.



Gambar 4.8 Perancangan Kategori OID

Pada Gambar 4.8 menjelaskan MIB berisi seperangkat Nilai, baik statistik maupun *control*. Untuk menyederhanakan MIB, File MIB adalah kumpulan Permintaan *Manager* kepada *agent. Agent* hanya mengumpulkan permintaan ini dan menyimpannya secara lokal dan menyiapkan respon ke *manager* saat diminta.

4.7 Perancangan Aplikasi Web

Dalam rancangan aplikasi web disini adalah tampilan pada sistem *monitoring* perangkat IoT. Dari hal ini aplikasi web dapat menerima atau memasukan data dan data juga tersimpan di database. Aplikasi web mempunyai fungsi masing-masing pada berikut ini :

1. Untuk menampilkan fitur Login.
2. Untuk menampilkan daftar *Agent* yang up atau down.
3. Untuk menghapus *Agent* yang tidak akan dilakukan *monitoring.*
4. Untuk memberhentikan atau menjalankan *Agent* yang di *monitoring*
5. Untuk menerima perintah dari user untuk membuat *monitoring.*
6. Untuk menyimpan data perintah ke dalam *database*.

7. Untuk menampilkan informasi dalam bentuk grafik.
8. Untuk melakukan filter informasi berdasarkan *host*, Kategori OID,Interval dan waktu komponen yang diamati.

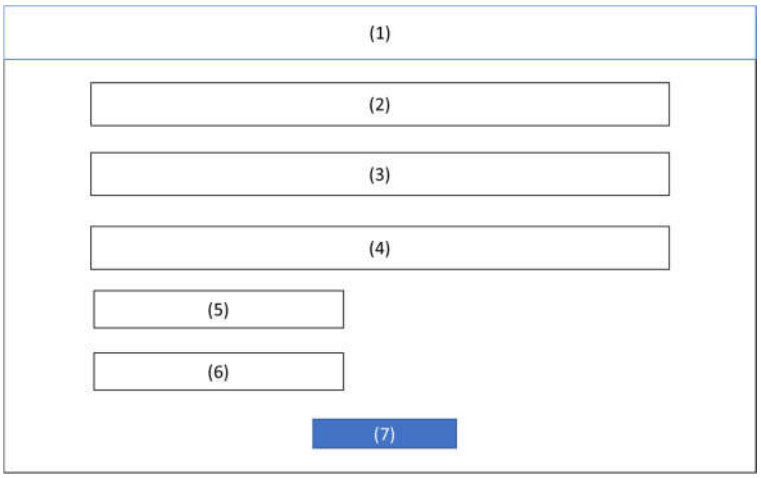
4.7.1 Rancangan antar muka aplikasi web

Rancangan antar muka halaman web yang ada adalah rancangan halaman index yang memiliki tugas untuk menampilkan daftar perangkat yang terhubung. Rancangan halaman berikutnya adalah rancangan halaman tambah monitor yang berfungsi untuk menambah perintah *monitoring*. Berikutnya adalah rancangan halaman hasil *monitoring* untuk menampilkan grafik hasil *monitoring*.

Gambar 4.9 Rancangan Halaman Index

Gambar 4.9 merupakan gambar yang menjelaskan dari rancangan halaman *Awal*, berikut penjelasan dari masing kegunaanya :

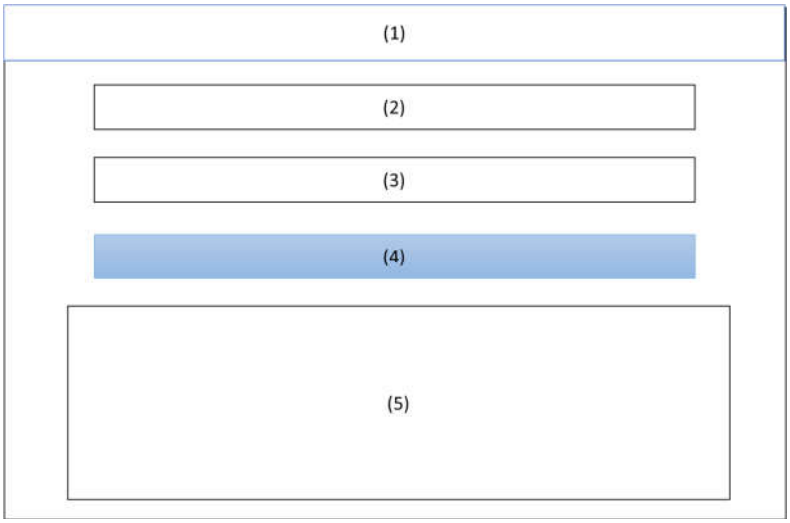
1. No. 1 pilihan navigasi dari sistem
2. No. 2 button *opsi monitoring phase,status device dll*
3. No. 3 tombol untuk melakukan monitor pada *device* yang dipilih
4. No. 4 button untuk menghapus *device* dari list *device*
5. No. 5 adalah daftar *Agent* yang terhubung pada *Manager*



Gambar 4.10 Rancangan Halaman Add Monitor

Gambar 4.10 ialah ditunjukan sebagai rancangan halaman Add monitor yang masing-masing mempunyai fungsi sebagai berikut :

1. No. 1 menu navigasi dari sistem *monitoring*
2. No. 2 form *hostname* yang ada secara otomatis di halaman utama
3. No. 3 form *monitoring kategori OID* untuk meminta pilihan *monitoring* apa yang akan dipilih
4. No. 4 form *interval* untuk meminta berapa jeda pengiriman waktu *monitoring*
5. No. 5 form date dan *start time* untuk meminta kapan waktu *monitoring* dimulai
6. No. 6 button *submit* untuk mengirimkan parameter yang sudah diisi



Gambar 4.11 Rancangan Halaman Hasil *Monitoring*

Gambar 4.11 ialah rancangan halaman result *monitoring* yang memiliki fungsi sebagai berikut :

1. No. 1 menu navigasi dari sistem *monitoring*
2. No. 2 pilihan host yang akan dipilih oleh user
3. No. 3 pilihan kategory *monitoring* yang akan dipilih oleh user
4. No. 4 button filter untuk melakukan filter berdasarkan parameter di atas.
5. No. 5 tampilan grafik hasil *monitoring*

4.8 Perancangan Database

Rancangan database disini menjelaskan bahwa data hasil *monitoring* akan tersimpan dengan benar. Dengan pentingnya rancangan database ini harus sesuaikan dengan tepat. Rancangan juga dibuatkan tabel dengan kebutuhan yang ada.

Petama disini ialah tabel host , untuk menyimpan data sot yang terhubung pada manager. Adapun Struktur tabel *host* dapat lihat dengan Tabel 4.1.

Tabel 4.1 Struktur Tabel Host

No	Nama Atribut	Tipe	Panjang	Keterangan
1	*Host*	TEXT	64	*MAC Address Host*
2	*Hostname*	TEXT	64	*IP Address Host*
3	*Status*	TEXT	10	Status Aktif/Tidak Aktif
4	*Phase*	TEXT	10	Status fase aktif *Monitoring*

kedua ialah tabel monitor yang berfungsi sebagai tabel yang menyimpan perintah *monitoring* dari manager kepada agent terlhat pada Tabel 4.2.

Tabel 4.2 Struktur Tabel *Monitor*

No	Nama Atribut	Tipe	Panjang	Keterangan
1	*Host*	TEXT	32	MAC *Address Host*
2	Monitype	INT	-	Tipe Data *Monitoring* (OID)
3	Interval	INT	-	Interval waktu *monitoring*
4	Waktu Mulai	INT	-	Waktu awal *monitoring*
5	Waktu Berakhir	INT	-	Waktu akhir *monitoring*

6	Waktu kirim	TEXT	64	Menyimpan waktu perintah dibuat

ketiga adalah tabel yang menyimpan data *resource*. Struktur data yang dikirimkan dalam proses *monitoring* juga sama dengan struktur data tabel yang ada. Struktur data ini dapat dilihat Tabel 4.3.

Tabel 4.3 Struktur Tabel *Resource*

No	Nama Atribut	Tipe	Panjang	Keterangan
1	*Host*	TEXT	64	*MAC Address Host*
2	Tanggal	TEXT	32	Tanggal data diambil
3	Waktu	TEXT	32	Waktu ketika *resource* diambil
4	*CPU usage*	INT	-	Penggunaan *CPU* keseluruhan dari *device*
5	*Memory Used*	INT	-	Memori yang digunakan
6	*Memory Available*	INT	-	Memori yang tersedia
7	*Swap*	INT	-	*Swap* dari *device*

Keempat adalah tabel yang menyimpan data *network*. Struktur data yang dikirimkan dalam proses *monitoring* juga sama dengan struktur data tabel *network* yang ada.Struktur ini dapat dilihat Tabel 4.4.

Tabel 4.4 Struktur Tabel Network

No	Nama Atribut	Tipe	Panjang	Keterangan
1	*Host*	TEXT	32	*MAC Address Host*
2	Tanggal	TEXT	32	Tanggal data diambil
3	Waktu	TEXT	32	Waktu ketika data diambil
4	*Byte Sent*	INT	-	Jumlah *byte* yang terkirim
5	*Byte Receive*	INT	-	Jumlah *byte* yang diterima
6	*Packet Sent*	INT	-	Jumlah *packet* yang dikirm
7	*Packet Receive*	INT	-	Jumlah *packet* yang diterima